"Proof-of-Work" Proves Not to Work

Ben Laurie[1] and Richard Clayton[2]

[1] ALD Ltd, The Stores, 2 Bath Road, London W4 1LT, United Kingdom
[2] University of Cambridge, Computer Laboratory, William Gates Building, 15 JJ Thompson Avenue, Cambridge CB3 0FD, United Kingdom
ben@algroup.co.uk, richard.clayton@cl.cam.ac.uk

Abstract. A frequently proposed method of reducing unsolicited bulk email ("spam") is for senders to pay for each email they send. Proof-of-work schemes avoid charging real money by requiring senders to demonstrate that they have expended processing time in solving a cryptographic puzzle. We attempt to determine how difficult that puzzle should be so as to be effective in preventing spam. We analyse this both from an economic perspective, "how can we stop it being cost-effective to send spam", and from a security perspective, "spammers can access insecure end-user machines and will steal processing cycles to solve puzzles". Both analyses lead to similar values of puzzle difficulty. Unfortunately, real-world data from a large ISP shows that these difficulty levels would mean that significant numbers of senders of legitimate email would be unable to continue their current levels of activity. We conclude that proof-of-work will not be a solution to the problem of spam.

1 Introduction

It is often suggested that unsolicited bulk email ("spam") is such a problem on the Internet because the current economic framework for email handling does little to discourage it. If only, it is suggested, the senders of email could be made to pay for their messages. Spammers would then cease their indiscriminate distribution of messages and email volumes would reduce as the senders targeted more carefully or just gave up altogether. Nevertheless, almost no one (other than those hoping for a handling fee) thinks that using actual money is a good way to achieve this economic utopia and even the holders of patents for "e-money" systems have failed to generate any significant enthusiasm for their wares.

However, there is an alternative to real-world money, which was first proposed by Dwork and Naor in 1992 [8]. Their idea was to have the sender of an email perform a complex computation as evidence that they believe that an email is worth receiving. The sender then proves to the recipient that this processing work has been completed and the email will then be accepted. The processing time is "free", so there is a minimal burden upon legitimate senders, but it is a finite resource, so that the spammers will not have unlimited amounts of processing time at their disposal and so cannot continue to send in bulk.

In this paper we briefly review "proof-of-work" systems in Section 2 and then in Section 3 we calculate the amount of work that should be proved. It

is inherent in a proof-of-work puzzle that it should take some time to compute and this will inevitably delay the sending of legitimate email. This paper seeks to answer the crucial question "how much time?". We present data on current email sending activity, which gives us sufficient figures to propose some values. In Section 4 we examine some real-world data to see how variability in actual usage of email fits in with the average case we have calculated. These calculations, developed here for the first time, show that it is currently impossible to discourage spammers sufficiently by means of a "proof-of-work" system without having an unacceptable impact on legitimate senders of email.

2 Proof-of-Work Systems

Dwork and Naor's 1992 scheme proposed a central authority that would hold a secret key, but decentralised systems would be far more practical on today's Internet. The most widely known proof-of-work system is Back's independently invented "hashcash" [1] which requires the sender to produce a string whose cryptographic hash starts with a certain number of zeroes[1]. An important property of the hashcash puzzle (and all proof-of-work puzzles) is that they are very expensive to solve, but it is comparatively cheap to verify the solution.

In order to tie the hashcash puzzle to a particular email, some parts of the string that must be produced are the email recipient address, a timestamp and a nonce. These preset values ensure that a puzzle solution is only valid for a particular destination and checks can be made to ensure that the solution is relatively recently constructed and has not been presented to the recipient before. The hashcash scheme is immune to "man-in-the-middle" attacks, whereby an innocent user is fooled into computing values for the benefit of another.

Proof-of-work puzzles have not been restricted to email, but have also been proposed for metering visits to websites [10], providing incentives in peer-to-peer systems [20], mitigating distributed denial-of-service attacks [17] and rate limiting TCP connections [15]. Jakobsson and Juels [14] considerably extend this list of potential usages. Wherever these systems are using proof-of-work as a way of limiting the abilities of the bad guys, the type of analysis we provide in this paper will be relevant.

It has always been an acknowledged problem with proof-of-work schemes that the amount of processing power available to particular users can vary enormously. Work that might take 10 seconds on a 3GHz Pentium could take an hour or more on a Palm Pilot. To address this problem, Dwork et al. [9] have recently proposed puzzles that rely on accessing large amounts of random access memory. Because memory speeds do not vary nearly as much as CPU speeds, these have far more constant performance across different machines – a factor of only four between slowest and fastest is claimed.

Finally, it should be noted that whilst most of the proposed proof-of-work schemes do not perform a useful calculation, Jakobsson and Juels [14] suggest

[1] In fact, the original version called for two hashes with the same initial bit-string – a single hash starting with zeroes was a recent improvement

the term "Bread Pudding Protocol" for any system whereby the results can be re-used for another purpose. Where this is occurring, this might make environmentalists feel slightly happier about the amount of power that is being consumed for no directly valuable purpose.

3 How Much Work Should You Prove?

We assume that the goal of introducing a proof-of-work system is to reduce the amount of spam the average person receives to below some fraction, S, of their legitimate email. Independent "consumer" testing of the best commercial spam filtering solutions shows that they currently achieve an S of 0.06 [22], but if one is going to rebuild the email infrastructure to incorporate proof-of-work one might hope to reduce S to 0.01 (1% of email is spam) or, given the significant effort and disruption involved in such a rebuilding enterprise, 0.001 (just one email in a thousand).

First of all, we need to know how much legitimate email each person receives. Radicati estimated [18] that, as of November 2003, 5.7×10^{10} emails are sent per day and quote a figure of 5.13×10^8 email users on the Internet using 9.02×10^8 email accounts. These numbers are growing rapidly: three months later [19] Radicati were estimating 5.74×10^8 users. Clearly these numbers are only estimates, but they are in line with those quoted by various other research organisations.

At the same time, November 2003, Brightmail were estimating that 56% of all email was spam [5] (other filtering companies give other figures, but in the same general range). This means that the daily total was 3.2×10^{10} spam and 2.5×10^{10} legitimate emails. Dividing this down by the population figure, we can see that each email user received, on average, about 50 legitimate and about 60 spam emails per day.

However, for our purposes, it is of more interest to consider computers rather than people since it is these computers that actually do the work that is to be proved. The Internet Domain Survey [13] provides an estimate of 2.3×10^8 hosts by counting how many systems have DNS entries. This means, given the figures above, that each machine is used by 2.5 people (and hence, on average, sends about 125 real emails). This figure of 125 is probably an overestimate, because we have ignored machines that are not directly connected to the Internet. We could address this inaccuracy by considering statistics on sales of PCs, firewalls or even Ethernet cards, but since we're about to make another sweeping generalisation we need not pick at this figure too hard.

We have assumed that the sending of legitimate email is equally distributed amongst all the Internet's users. This is the "best-case scenario" where the effort of providing proof-of-work falls equally upon everyone. However there is an obvious exception to this. A great deal of email is part of a "mailing list", viz: one email is sent to a "list exploder", which then distributes the email to many list subscribers. Since it is clearly impractical to calculate proof-of-work for each recipient of a mailing list, we assume that the original sender does a proof-of-work for delivery to the exploder and the recipients delegate the checking and

accept everything they are sent. Although the need to do this delegation may be obvious for community lists where everyone can contribute their email opinions, where the information flow is entirely one way (for example, a cinema's weekly 'What's On' summary), it may be less obvious to recipients that delegation is necessary. In this latter case the use of distributed trust systems, such as IronPort's "Bonded Sender" system [2] may have a rôle to play.

We were unable to locate any published data for mailing list volumes, for any type of list. Therefore we examined data for incoming email at a large (200,000 customer) ISP in the United Kingdom and counted (after a spam filtering stage) the volume attributable to senders who sent email to more than ten or more customers. We estimate that the proportion of non-spam email that is some kind of mailing list email is currently about 40%. Hence, assuming that the rest of the email is evenly distributed, this leaves us with a final average of about 75 legitimate non-list emails being sent by each host per day.

Therefore, an overview of our task is that we wish to set a cost of each proof-of-work of C so that it is possible to send an average of 75 emails per day whilst limiting the total amount of spam to $S \times 2.5 \times 10^{10}$ per day, for a value of S of at most 0.01 and preferably much less. However, we must also allow for the variable speed of hosts at solving puzzles, adjust for the amount of time that hosts are switched on (not necessarily online) per day and, since the 75 is only an average, allow for considerable variation in the amount being sent.

3.1 The Economic Approach

The classic method of estimating C is to express it in money terms and compare it with the profit for each spam email. Goodman and Rounthwaite [12] provide a survey of how much professional spammers charge for sending emails. They give examples from 0.03 to 0.001 cents per email, and one might deduce from their information that below 0.005 cents is only marginally profitable.

If we assume a standard spam-capable PC unit (no monitor is needed) costs around $500 and will last for a thousand days (almost 3 years), then the marginal cost per day is 50 cents. It will also cost about another 25 cents per day in electricity (120 watts, 8.5 cents/kWh). Spam emails are generally small, so the cost of connectivity can be amortised away over many machines. Hence, with a total cost of 75 cents each day, the spammer can cover these costs by sending 15,000 emails per 24 hours at 0.005 cents each. Therefore, C must be set so that each calculation takes at least 5.8 seconds. Naturally, using special-purpose hardware (e.g. FPGAs) instead of standard PCs could well reduce the cost per email of creating the proof-of-work, and therefore proportionately increase C.

Although spamming operations can be highly automated, the spammer will wish to be paid for their labour. If they were in the business full-time they might be looking for at least $100/day (c. $30K/annum) and if they operated 100 machines (a $50K investment) they'd be looking to clear 100 cents/day per machine over and above the running costs. Viz: they'd be looking to send 35,000 emails per 24 hours per machine at 0.005 cents each, (i.e.: 3.5 million emails per

day – a quite plausible number: in April 2004, Scott Richter is reported to be sending $50 - 250$ million messages a day, charging 0.02 cents each [3]).

According to Goodman and Rounthwaite's pricing survey, spammers used to charge as much as 0.1 cents per email. If the result of deploying proof-of-work systems were to be that the price returned to this amount then it would be necessary to restrict the spammer to 1,750 emails per day, by setting the calculation time to at least 50 seconds.

The price that spammers can charge the advertisers that use their service will depend on how cost-effective spam is, which will depend upon the response rate that is achieved. Figures for response rates to legitimate "opt-in" email show responses to sales promotions varying from 0.7% to 1.6% over time [24]. Actual figures for response rates to spam emails are rare, although there is a lot of "what if" speculation. In 2002 the Wall Street Journal [16] gave real-world examples of spam response rates of 0.013% and 0.0023%. If the rate per email did indeed return to 0.1 cents then at a 0.0023% response rate then advertisers would need to be selling goods with a profit margin of at least $4.35. This is not implausible: mortgage leads are worth $50, cellphone sales about $85 and there are examples of companies selling fake medicines worth $2.50 for $59.95 [7].

Hence we would suggest that a price of 0.1 cents per email could return, especially if the senders of spam were to improve response rates by learning lessons in presentation from legitimate marketers. Thus we must look to restrict spammers to just 1,750 emails per day per machine. Of course this would also limit legitimate senders to the same level of sending. Since, as we calculated above, the average machine will only need to send about 75 emails, this gives some "headroom" for legitimate activity, but – when one starts to consider variations in sending activity and differences in puzzle-solving speeds – not all that much.

Unfortunately, this method of estimating C does not give much insight into the value of S (how much spam will continue to arrive). The difficulty is that the effect will be to suppress the least profitable forms of spam and we have no idea what proportion this might be. However, if spam is more convincingly presented, or the spammer can make a profit from not only emailing but owning other parts of the supply chain, then it will certainly not be possible to raise C enough to have an impact without starting to affect legitimate email.

3.2 The Free CPU Approach

A more realistic approach is to realise that spammers will not necessarily be purchasing their own hardware. Because email traffic from their own machines is now widely blocked, they relay a great deal of their spam via poorly administered third-party machines. In the past, insecure email servers were exploited. Today, it is far more likely that an incorrectly configured HTTP or SOCKS proxy will be used to access an email server that trusts the insecure machine. On a typical day in February 2004, the SORBS DNS Blocklist [23] listed 960,000 open HTTP proxies and 1,200,000 open SOCKS proxies. Most could be used to relay email.

A more recent development has been for spammers to take over (or "*0wn*" in hacker parlance) the machines. Many recent viruses have carried a payload that

would allow a remote person to control the infected machine. ICSA Labs gives a 2003 figure for virus infections in medium to large US companies of 108 machines per thousand per year [4], but virus protection differs markedly in other sectors. Estimates of global virus infection rates seem to be little more than guesses, but in late January 2004 MyDoom.a was reported by F-Secure to have infected a million machines with NAI putting it at half that figure. At the large UK ISP already mentioned one customer in 200 was infected by MyDoom – logs of their outgoing email tripped pattern recognition heuristics [6] – and this rate would scale to 1.1 million machines across the whole Internet.

At present, infections by mass-mailing viruses such as MyDoom are relatively easy for third parties to detect. The systems send out lots of copies of the virus, which will be reported by recipients or may be detected by ISP systems. A virus where many of the infected machines just kept quiet, and calculated "proof-of-work" results for others, would be invisible to third parties. It would also be invisible to the machine owner if the appropriate steps were taken to keep the calculations at a lower operating system priority than other tasks, the approach used for background calculations by systems such as SETI@home [21]. Hence, undetectability will allow a large number of machines to be *0wned*.

If we assume that spammers switched all of their email to these *0wned* machines (each gaining control of a suitable proportion) then, to maintain November 2003 sending rates, a pool of a million machines would have to send 32,000 emails each per day. This value is consistent with the UK ISP's currently observed average number of emails sent by exploited machines of 21,000 per day.

If we require S to be 0.01, for example, then we need to reduce the amount of spam sent to 250 emails per day, with a value for C of 346 seconds, not a long way from the figure we arrived at using the economic argument above. Of course, if the spammers manage to *0wn* more than a million machines, or we require S to be 0.001, then C must be increased accordingly.

4 So What Might Be Practical?

We examined logging data from the large UK ISP for those customers who use the outbound "smarthost" (about 50,000 on this particular weekday). We excluded the one customer being actively exploited by a spammer and the four infected by a virus; and also the 145 with configuration problems that caused email to loop continuously. Counts were made of the number of emails sent with particular SMTP "MAIL FROM" settings (which can be, to a first approximation, assumed to map to different individual senders using different individual machines). The cumulative frequency of total emails sent is plotted in Figure 1.

As can be seen, although 93.5% of machines sent less than the global average of 75 emails per day, the distribution has a very long tail. In particular, at the values calculated above of 1,750 and 250 emails per day, a proof-of-work scheme would prevent legitimate activity by 0.13% and 1.56% of customers respectively. Some of these will be operating mailing lists and may not need to provide proof-of-work, but other eBusiness systems will surely be caught.

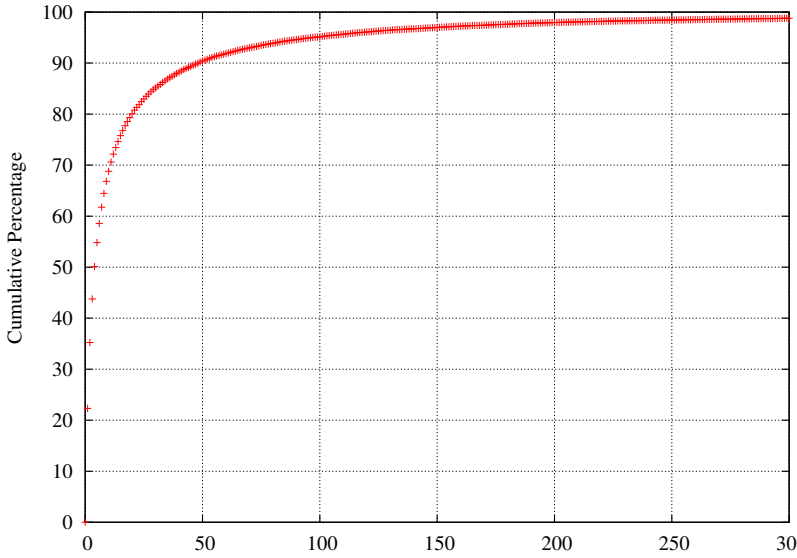

Fig. 1. Senders of x emails per day

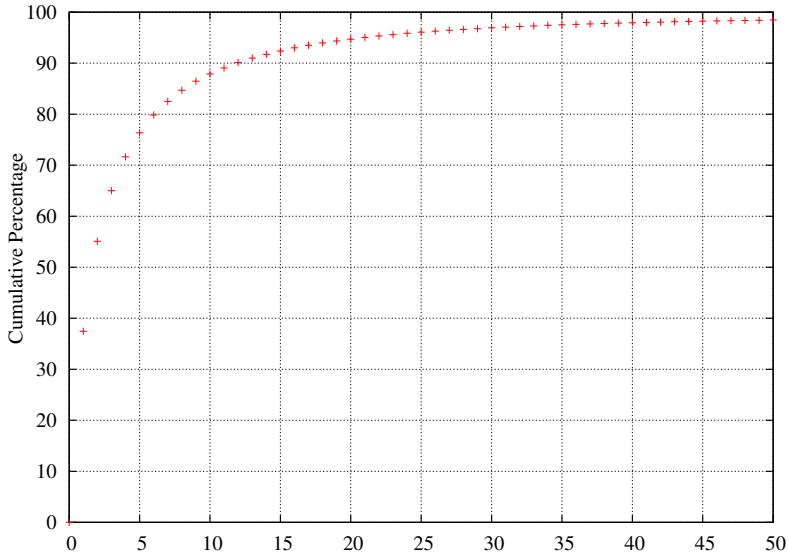

Fig. 2. Senders of x emails per hour

The position is considerably worse if we consider hourly sending rates as presented in Figure 2. We plot how many emails each customer sent during any hour that they were active. We must assume that the spammers will be running 24 hours a day, so we need to restrict them, by the calculations above, to 73 or 11 emails per hour. However, many users will only switch on their machines shortly before they actually send their email. As can be seen, the proof-of-work scheme would now inconvenience 1% or 13% of legitimate email users.

5 Conclusions

We have presented two different methods of calculating the limits to be placed on the sending of spam. If we try to make it uneconomic to send spam then we must restrict senders to 1,750 messages a day. Alternatively, the limit must be 250 messages a day if we want to reduce spam to 1% of normal email, but assume that the spammers can steal the efforts of a million compromised machines.

None of our analysis goes into the nature of the puzzles that need to be solved, nor in absolute terms how fast they can be tackled. However, the spammers will select "fast" machines and real senders may only have "slow" machines, so for comparison purposes, the 1,750 value should be set lower by the factor of four in solving-time variation that the best known puzzle systems currently achieve.

Unfortunately, the limits we would wish to set are of the same order of magnitude as global averages for usage. Not surprisingly therefore, examining the variations of email sending by some real users showed that significant numbers of them would be unable to provide "proof-of-work" for all the legitimate email that they currently send, making the system completely infeasible to deploy.

We conclude that proof-of-work, on its own, it is not a panacea for fighting spam unless the spammer's cost base can be significantly increased and the number of insecure end-user machines is significantly reduced. For proof-of-work schemes to be plausible one would be looking for many orders of magnitude between the work done by the "good guys" and that achievable by the "bad guys". We are also concerned that some of the other situations where proof-of-work has been proposed have not been properly analysed to consider how much money the bad guys can spend and how many resources they might steal.

What might work against spam? Systems based on human effort ("captchas") have different types of costs than cryptographic puzzles. It is also possible that a hybrid scheme will be of value, where proof-of-work is only used occasionally. For example, Gabber et al [11] have a scheme for creating "extended email addresses" which can be revoked when abused. We leave analysis of such schemes to future work – fully expecting to be able to debunk some further proposals by the simple application of real-world measurements and values.

Acknowledgements

We wish to thank the anonymous reviewers for reminding us to include calculations for spammer's salaries and for their numerous other helpful comments. We

also wish to acknowledge the co-operation of Demon Internet in making ISP data available, and the financial assistance provided by the Cambridge MIT Institute (CMI) to Richard Clayton through the project: "The design and implementation of third-generation peer-to-peer systems".

References

1. A. Back: Hashcash. 1997. http://www.cypherspace.org/adam/hashcash/
2. S. Bannister: Bonded Sender Program Overview. IronPort Inc, Jul 2002.
3. P. Boutin: Interview with a Spammer. InfoWorld, 16 Apr 2004.
4. L. Bridwell: ICSA Labs 9th Annual Computer Virus Prevalence Survey. ICSA Labs, 2004.
5. Brightmail Inc: Spam Percentages and Spam Categories, Feb 2004. http://www.brightmail.com/spamstats.html
6. R. Clayton: Stopping Spam by Extrusion Detection. To appear, 2004.
7. S. Cobb: The Economics of Spam. ePrivacy Group, Feb 2003.
8. C. Dwork and M. Naor: Pricing via Processing or Combatting Junk Mail. in E. F. Brickell (Ed.): Advances in Cryptology – CRYPTO '92, LNCS 740, Springer Verlag 1992, pp.139–147.
9. C. Dwork, A. Goldberg and M. Naor: On Memory-Bound Functions for Fighting Spam. in D. Boneh (Ed.): Advances in Cryptology – CRYPTO 2003, LNCS 2729, Springer Verlag 2003, pp. 426–444.
10. M. K. Franklin and D. Malkhi: Auditable Metering with Lightweight Security. in: Financial Cryptography, 1997, pp. 151–160.
11. E. Gabber, M. Jakobsson, Y. Mathias and A. Mayer: Curbing Junk E-Mail via Secure Classification. in : Financial Cryptography, 1998, pp. 198–213.
12. J. Goodman and R. Rounthwaite: Stopping Outgoing Spam. ACM Conference on Electronic Commerce, EC'04, 2004.
13. Internet Systems Consortium: Internet Domain Survey, 2004. http://www.isc.org/ops/ds/reports/2004-01/
14. M. Jakobsson and A. Juels: Proofs of Work and Bread Pudding Protocols. in: Proceedings of the IFIP TC6 and TC11 Joint Working Conference on Communications and Multimedia Security (CMS '99). Kluwer, 1999.
15. A. Juels and J. Brainard: Client puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks. in: Proceedings of NDSS '99 (Networks and Distributed Systems Security), 1999, pp. 151–165.
16. M. Mangalindan: For Bulk E-Mailer, Pestering Millions Offers Path to Profit. Wall Street Journal, 13 Nov 2002.
17. D. Mankins, R. Krishnan, C. Boyd, J. Zao and M. Frentz: Mitigating Distributed Denial of Service Attacks with Dynamic Resource Pricing. in: Proceedings of 17th Annual Computer Security Applications Conference (ACSAS 2001), 2001.
18. Radicati Group Inc.: Market Numbers Quarterly Update, Q4 2003.
19. Radicati Group Inc.: Market Numbers Quarterly Update, Q1 2004.
20. A. Serjantov and S. Lewis: Puzzles in P2P Systems. 8th CaberNet Radicals Workshop, Corsica, Oct 2003.
21. SETI@home: http://setiathome.ssl.berkeley.edu/
22. J. Snyder: Test: Spam in the wild. Network World Fusion, 9 Sep 2003. http://www.nwfusion.com/reviews/2003/0915spam.html
23. SORBS: Spam and Open Relay Blocking System. http://www.sorbs.net
24. R. Wussler: Building High Response E-Mail. Harte Hanks, Jan 2004.